Systems Management 93-614

MULTI-TARGET TRACKING - PHASE II

TASK II - DETAIL DESIGN REPORT

DSS Contract No.:	003ST.W7714-2-9620
Submission Date:	December, 1992
Thomson Systems File No.:	G0649C
Revision:	Final

THOMSON-CSF
SYSTEMS CANADA

SYSTEMS CANADA

MULTI-TARGET TRACKING - PHASE II

TASK II - DETAIL DESIGN REPORT

Prepared by:

Banu Kalaycioglu
Systems Engineer

Approved by:

A. Kurkjian
Azaduer Kurkjian
Project Manager

Thomson-CSF Systems Canada Inc.
18 Auriga Drive
Ottawa, Ontario K2E 7T9

DSS Contract No.:	003ST.W7714-2-9620
Submission Date:	December, 1992
Thomson Systems File No.:	G0649C
Revision:	Final

MASTER DOCUMENT REVISION LEVEL SHEET

REVISION	COMMENTS	DATE	BY
Draft		NOV 92	BK
Final		DEC 92	BK

P148371.PDF [Page: 6 of 61]

Table Of Contents

		Page
1	SCOPE	1
1.1	ATP Package Overview	1
1.2	Document Overview	1
2	REFERENCED DOCUMENTS	2
3	DETAILED DESIGN	2
3.1	Baseline - ATP Modifications	2
3.1.1	FRAME Routine	3
3.1.2	INIT1 Routine	3
3.1.3	RSIM Routine	4
3.1.4	CORR Routine	4
3.1.5	DISP Routine	4
3.2	Hough Transform Track Initiation	4
3.2.1	INIT1 Routine Design	4
3.2.1.1	Description	4
3.2.1.2	Inputs	5
3.2.1.3	Outputs	5
3.2.1.4	Global Data	5
3.2.1.5	Local Data	6
3.2.1.6	Main Body of the Routine	6
3.2.2	COPY_RET_TO_OLD_RETURN Routine Design	6
3.2.2.1	Description	6
3.2.2.2	Inputs	7
3.2.2.3	Outputs	7
3.2.2.4	Global Data	7
3.2.2.5	Local Data	7
3.2.2.6	Main Body of the Routine	8
3.2.3	FIRST_PASS_DETECTION Routine Design	8
3.2.3.1	Description	8
3.2.3.2	Inputs	9
3.2.3.3	Outputs	9
3.2.3.4	Global Data	9
3.2.3.5	Local Data	10
3.2.3.6	Main Body of the Routine	10

P148371.PDF [Page: 7 of 61]

Table Of Contents

		Page
3.2.4	SECOND_PASS_DETECTION Routine	12
3.2.4.1	Description	12
3.2.4.2	Inputs	12
3.2.4.3	Outputs	13
3.2.4.4	Global Data	13
3.2.4.5	Local Data	14
3.2.4.6	Main Body of the Routine	15
3.2.5	BUILD_HISTOGRAM Routine Design	16
3.2.5.1	Description	16
3.2.5.2	Inputs	17
3.2.5.3	Outputs	17
3.2.5.4	Global Data	17
3.2.5.5	Local Data	18
3.2.5.6	Main Body of the Routine	18
3.2.6	DETECT_TARGET Routine Design	19
3.2.6.1	Description	19
3.2.6.2	Inputs	20
3.2.6.3	Outputs	20
3.2.6.4	Global Data	21
3.2.6.5	Local Data	21
3.2.6.6	Main Body of the Routine	22
3.2.7	INITIATE_TRACK Routine Design	22
3.2.7.1	Description	22
3.2.7.2	Inputs	23
3.2.7.3	Outputs	23
3.2.7.4	Global Data	23
3.2.7.5	Local Data	24
3.2.7.6	Main Body of the Routine	24
3.3	Logic-Based Track Initiation	25
3.3.1	INIT1 Routine Design	25
3.3.1.1	Description	25
3.3.1.2	Inputs	26
3.3.1.3	Outputs	26
3.3.1.4	Global Data	26
3.3.1.5	Local Data	28
3.3.1.6	Main Body of the Routine:	28

P148371.PDF [Page: 8 of 61]

Table Of Contents

		Page
3.3.2	INITIATE_TRACKS Routine Design	30
3.3.2.1	Description	30
3.3.2.2	Inputs	30
3.3.2.3	Outputs	30
3.3.2.4	Global Data	30
3.3.2.5	Local Data	31
3.3.2.6	Main Body of the Routine	32
3.3.3	SAVE_CONTACT Routine Design	34
3.3.3.1	Description	34
3.3.3.2	Inputs	34
3.3.3.3	Outputs	34
3.3.3.4	Global Data	35
3.3.3.5	Local Data	35
3.3.3.6	Main Body of the Routine	35
3.3.4	ASSOCIATION_PROC Routine Design	35
3.3.4.1	Description	35
3.3.4.2	Inputs	36
3.3.4.3	Outputs	36
3.3.4.4	Global Data	37
3.3.4.5	Local Data	37
3.3.4.6	Main Body of the Routine	39
3.3.5	UPDATE_TRACK Routine Design	41
3.3.5.1	Description	41
3.3.5.2	Inputs	41
3.3.5.3	Outputs	41
3.3.5.4	Global Data	41
3.3.5.5	Local Data	42
3.3.5.6	Main Body of the Routine	43
4	ATP GLOBAL DATA	47
4.1	Baseline - ATP Modifications	47
4.2	Hough Transform Track Initiation	47
4.3	Logic_based Track Initiation	48
5	ATP DATA FILES	50
5.1	CASE.DAT File	50
5.2	Radar Data File	50

P148371.PDF [Page: 9 of 61]

1 SCOPE

The main objective of this phase of the project is to investigate alternative track initiation techniques which can be simulated and tested in the context of the ATP software environment. As a result of this work, the new track initiation modules will be developed and integrated in the framework of the ATP package. The performance of the new track initiation algorithms in a clutter environment will be evaluated and documented in the form of a report containing the simulation output and results of the analysis.

1.1 ATP Package Overview

The Radar Division of DREO has developed Multi-Target Tracking software referred to as Automatic Tracking Program (ATP) package, in order to test the different tracking algorithms by using the real time radar data.

The ATP software package is written in VAX FORTRAN and is built around a framework program (FRAME) which calls all the necessary routines for complete automatic tracking operations. The routines called by the FRAME program include:

(a) Tracking Operation Set Up (SETUP);
(b) Radar Data Input (RSIM);
(c) Plot-to-Track Association (CORR);
(d) Track Initiation / Termination (INIT1);
(e) Filtering (FILT);
(f) Display (DISP); and
(g) Tracking Statistics (LOG).

1.2 Document Overview

This document describes the complete design specification for the Hough Transform and Logic-Based Track Initiation Techniques and also the modifications for the Baseline ATP package in order to implement these algorithms.

Section 2 lists the referenced documents.

Section 3.1 describes the modifications to the original ATP package to create a baseline software which provides a compatible interface for the different track initiation routines (INIT1) to be implemented.

P148371.PDF [Page: 10 of 61]

Section 3.2 describes the Hough Transform Track Initiation Technique and the detailed design of its routines.

Section 3.3 describes the Logic-Based Track Initiation Technique and the detailed design of its routines.

In Section 4, the global data for the two new track initiation techniques are discussed separately. This section also describes the new global structures added to the ATP program due to the ATP baseline modifications.

Section 5 discusses the ATP data files and finally Section 6 documents the Requirements Traceability matrix.

2 REFERENCED DOCUMENTS

(1) Proposal for Multi-Target Tracking - Phase II, Thomson Systems, 25 June, 1992;

(2) Multi-Target Tracking DSS Contract No. 003ST.W7714-2-9620;

(3) Raid Tracking Trials (RATT): Digital Plot Data Base", DREO Report No. 1026, March 1990;

(4) Automatic Tracking Programs (ATP) Software Package On The Radar VAX Computer, Radar Research Laboratory, CRC, April 1987;

(5) Multi-Target Tracking with Radar Application, S. Blackman;

(6) Multi-Target Tracking - Phase II, Task 1 Report, Thomson Systems, November 1992.

3 DETAILED DESIGN

3.1 Baseline - ATP Modifications

In order to use the different track initiation techniques, without changing the other modules or the interfaces, the data structures used in the FRAME/INIT1 interface must be compatible for all the track initiation techniques.

P148371.PDF [Page: 11 of 61]

The original ATP package uses A and B arrays interchangeably to store the returns from the current scan and the returns from the previous scans. The new track initiation techniques require different array structures and sizes to store the radar data from the previous scans. For the sake of compatibility, in the modified version of the ATP package, A array will be dedicated to the current returns where B (will be renamed as OLD_RET) keeps only the returns from the previous scans. OLD_RET array will be defined as global and the declaration will change with the usage of different track initiation techniques. This will be accomplished by creating different include files for different initiation routines (Hough Transform and Logic-Based Initiation) and including the appropriate file in the INIT1 and other initiation routines.

These modifications are explained in more detail in Section 2.2 of the Software Specification Document for Task I. These changes are considered as modifications to the existing ATP package rather than a new design. Therefore, the original design of the ATP routines are not discussed here. Only the modified areas of the certain ATP routines effected by these changes are explained.

3.1.1 FRAME Routine

The definition of A array which keeps the returns from the current scan will be changed from (0:50, 50,10) to (0:50,15) where A(0,1), A(i,j) and A(i,10) correspond to A(0,0,1), A(i,0,j) and A(i,1,1) in the old definition respectively. (i > 0 and j changes from 1 to 9).

The definition of the B array will remain the same. However, this array will be declared as global and the name will be changed to OLD_RET array. Only A array will be used to store the data for the current scan and OLD_RET array will be dedicated to the returns from the previous scan. OLD_RET array will be defined as (0:50,11). Note that location 11 contains the scan number of the return. The sections of the code which refer to the A & B arrays will be modified according to the above changes.

3.1.2 INIT1 Routine

This routine will convert the A(0:50,15) format to A(0:50,50,10) format before the track initiation process. After the track initiation process, the current returns will be copied to OLD_RET table for the next scan track initiation.

P148371.PDF [Page: 12 of 61]

3.1.3 RSIM Routine

The sections of the code which refer to the A and B arrays will be modified according to the changes in section 3.1.1.

3.1.4 CORR Routine

The sections of the code which refer to the A and B arrays will be modified according to the changes in section 3.1.1.

3.1.5 DISP Routine

The sections of the code which refer to the A and B arrays will be modified according to the changes in section 3.1.1.

3.2 Hough Transform Track Initiation

The Hough Transform Track Initiation Technique is discussed in Section 2.2.2 of the Software Specification Document for Task I of the MTT Phase II project. The inputs, outputs and the algorithm are also described in Section 2.2.2 in a general sense.

Following sections document the detailed design of the routines to be implemented for the Hough Transform Track Initiation Technique.

3.2.1 INIT1 Routine Design

3.2.1.1 Description

This routine is the control routine for the track initiation process. It calls the appropriate routines each time it is invoked. There is an "n" number of (will be specified later) scans of waiting period. The purpose of this period is to store the n scans of plot data in the OLD_RET array to be used in the initiation process. When this period is over, the FIRST_PASS_DETECTION routine is called. This routine builds the first histogram with a coarse quantization and then identifies the regions which require a more detailed quantization. The results of the first pass are fed into the SECOND_PASS_DETECTION routine where the actual tracks are started.

P148371.PDF [Page: 13 of 61]

The SECOND_PASS_DETECTION routine builds another histogram with a higher resolution for the areas identified by the FIRST_PASS_DETECTION routine and it starts the tracks for the identified bins of the second histogram.

This routine is only called by FRAME module.

3.2.1.2 Inputs

REAL RETURNS(0:LIM_RET, 15):
where LIM_RET is a user defined global parameter. This array contains the information for all the returns from the current scan. The RETURNS array is explained in Sec. 3.1.1 as A array.

REAL NEW_TRACKS(LIM_RET, 7):
where LIM_RET is a user defined global parameter. This is a temporary array used to pass the information about the new tracks created by INIT1 routine to the ADD_TRACKS routine.

3.2.1.3 Outputs

REAL NEW_TRACKS(LIM_RET, 7): see Inputs above.

3.2.1.4 Global Data

INTEGER SCAN_COUNT:
This global variable is set to 0 after every initiation process and it is used to store the plot data for a defined number of scans during the waiting period (to save the n scans of data in the OLD_RET array).

INTEGER MAX_NUMBER_OF_SCANS:
The literal value for the maximum number of scans of radar data to be used in the initiation process. The actual value will be determined later.

REAL OLD_RET(0:MAX_NUMBER_OF_SCANS*LIM_RET, 11):
This array keeps the returns from a number (MAX_NR_OF_SCANS) of previous scans. It has the same structure as RETURNS array explained in Section 3.1.1 as A array.

P148371.PDF [Page: 14 of 61]

3.2.1.5 Local Data

REAL DEL_THETA1:
The parameter spacing value for the first phase of the detection (This value is chosen as (10*PI/180) Rad.

REAL DEL_THETA2:
The parameter spacing value for the second phase of the detection (This value is chosen as (1.0*PI/180) Rad.

3.2.1.6 Main Body of the Routine

```
SET DELTA_THETA1 TO 10.0*PI/180 Rad
SET DELTA_THETA2 TO 1.0*PI/180 Rad

INITIALIZE THE LOOK_UP_TABLE (See Section 4.2)

/*      Process the Start up period for MAX_NR_OF_SCANS scans, without actually doing any
        track initiation process, just copying the returns to the old_return array and
        incrementing the number of scans processed */

INCREMENT SCAN_COUNT BY 1

CALL COPY_RET_TO_OLD_RETURNS(RETURNS)

/*      if the scan_count is greater than the MAX_NR_OF_SCANS, we can start the track
        initiation process */

IF (SCAN_COUNT >= MAX_NR_OF_SCANS) THEN

        CALL FIRST_PASS_DETECTION(DELTA_THETA1)
        CALL SECOND_PASS_DETECTION(NEW_TRACK, DEL_THETA1, DEL_THETA2)
        SET SCAN_COUNT TO 0

        /*      set the number of the returns saved in the OLD_RET table to zero */

        SET OLD_RET(0,1) TO 0

END IF
```

3.2.2 COPY_RET_TO_OLD_RETURN Routine Design

3.2.2.1 Description

This routine copies the plots from the current scan to the OLD_RET array. The most recent scan is located at the end of OLD_RET array. Column 10 of the OLD_RET array shows the validity of the stored radar data.
The number of returns stored in the OLD_RET array is kept in location (0,1) of the OLD_RET array.

This routine is called by INIT1 routine.

P148371.PDF [Page: 15 of 61]

3.2.2.2 **Inputs**

REAL RETURNS(0:LIM_RET, 15):
where LIM_RET is a user defined global parameter. This array contains the information for all the returns from the current scan. The RETURNS array is explained in Sec. 3.1.1 as A array.

3.2.2.3 **Outputs**

REAL OLD_RET(0:MAX_NUMBER_OF_SCANS*LIM_RET, 11):
This array keeps the returns from a number (MAX_NR_OF_SCANS) of previous scans. It has the same structure as RETURNS array explained in Section 3.1.1 as A array.

3.2.2.4 **Global Data**

INTEGER SCAN_COUNT:
This global variable is set to 0 after every initiation process by INIT1 routine. This variable is used to store the number of scans during the waiting period.

REAL OLD_RET(0:MAX_NUMBER_OF_SCANS*LIM_RET, 11):
This array keeps the returns from a number (MAX_NR_OF_SCANS) of previous scans. It has the same structure as RETURNS array explained in Section 3.1.1 as A array.

INTEGER MAX_NR_OF_SCANS:
Indicates the maximum number of scans for the waiting period before the Hough Transform Initiation

3.2.2.5 **Local Data**

INTEGER ROW, COL:
These variables are used for indexing into the RETURNS and OLD_RET arrays.

INTEGER OLD_INDEX:
This variable shows the start location of the current scan in the OLD_RET array

P148371.PDF [Page: 16 of 61]

3.2.2.6 Main Body of the Routine

```
/*      Copy the plots from the current scan to the OLD_RET array, OLD_RET(0,1)=nr of returns
        saved in the OLD_RET table */

OLD_INDEX = OLD_RET(0,1)

DO ROW = 1 TO RETURNS(0,1)

        /* Copy only the unused plots from the current scan */

        IF (RETURNS(ROW,10) = 0.0) THEN

                OLD_INDEX = OLD_INDEX + 1
                DO COL = 1 TO 11
                        SET OLD_RET(OLD_INDEX, COL) TO RETURNS(ROW,COL)
                END DO

        END IF

END DO

/*      Store the final size of OLD_RET table */

OLD_RET(0,1)=OLD_INDEX
```

3.2.3 FIRST_PASS_DETECTION Routine Design

3.2.3.1 Description

This is the control routine for the first phase of the track initiation. Appropriate routines are called to complete the initial track detection. The parameter spacing used for this phase is coarser than the one used for the second phase. FIRST_PASS_DETECTION routine first calls the routine BUILD_HISTOGRAM to create the histogram for the given quantization value then calls the routine DETECT_TARGET to identify the areas which need the further quantization in the second phase of the initiation.

If a target is detected in the bin the lower limit for the theta value and the number of bins for the second phase histogram are calculated and marked in the SEC_PASS_DATA array.

If the target is detected in bin number 1 or 18 then two intervals are magnified. Otherwise the number of intervals to magnify is three. (i.e., n-1, n, n+1). In this case the theta value (lower limit) is set to the theta value which corresponds to the (n-1)th bin.

This routine is called by only INIT1 routine.

P148371.PDF [Page: 17 of 61]

3.2.3.2 **Inputs**

REAL DEL_THETA1:
The parameter spacing value for the first phase of the detection (This value is chosen as (10*PI/180) Rad.

3.2.3.3 **Outputs**

REAL HIST(0:LIM_RET*MAX_NR_OF_SCANS):
The histogram bin, rows corresponding to the returns. This array (i.e. bin) keeps the rho values of each return for a particular theta value.

3.2.3.4 **Global Data**

REAL PI:
Literal value for the 180 degrees

INTEGER MAX_NR_OF_SCANS:
This variable keeps the literal value for the total number of scans to be stored in the OLD_RET array in order to invoke the initiation process.

REAL HIST(0:LIM_RET*MAX_NR_OF_SCANS):
The histogram bin, rows corresponding to the returns. This array (i.e. bin) keeps the rho values of each return for a particular theta value.

REAL VECT(LIM_RET*MAX_NR_OF_SCANS):
Used in grouping of the returns that belong to the parallel tracks. See Section 4.2.

REAL LOOK_UP(4,0:8):
Used in grouping of the returns that belong to the parallel tracks. See Section 4.2.

REAL SECOND_PASS_DATA(18,2):
Keeps the data for the identified bins by the first pass detection routine

INTEGER NR_OF_SEC_PASS_DATA:
Keeps the number of identified bins in the first pass detection phase

P148371.PDF [Page: 18 of 61]

3.2.3.5 Local Data

REAL THRESHOLD_VOTE, THRESHOLD_ERR:
The threshold values used to determine the target presence (error threshold and vote threshold)

INTEGER THRESHOLD_LEV:
Indicates the maximum order of averaging.

INTEGER NR_OF_BINS:
The number of the histogram bins

REAL LOWER_LIMIT:
The initial angle theta, used in building the histogram

INTEGER I, BIN, INDEX,RET
Used in the looping and indexing into the arrays.

INTEGER COUNT:
Temporary variable which keeps the number of returns belong to the same group.

3.2.3.6 Main Body of the Routine

```
/*      Following are the constant values used in the track detection; THRESHOLD_ERR is the
        threshold value for the diatance from the average, THRESHOLD_VOTE is the number of
        'votes' required for track detection process*/

DEFINE THRESHOLD_ERR AS 2000
DEFINE THRESHOLD_VOTE AS 4.0
DEFINE THRESHOLD_LEV AS 3

/*      The LOWER_LIMIT and the nr_of_bins values are passed to BUILD_HISTOGRAN Procedure. For
        the first phase of the track initiation the quantization of the theta starts at 0
        degree. */

SET LOWER_LIMIT TO 0
SET NR_OF_BINS TO PI/DEL_THETA1

/*      The histogram is built one bin at a time, and for each bin the presence of the target
        is checked */

DO BIN=1 TO NR_OF_BINS
        CALL BUILD_HISTOGRAM(BIN, LOWER_LIMIT, DEL_THETA1)
        CALL DETECT_TARGET(BIN,THRESHOLD_LEV)

        SET I TO 1
        SET INDEX TO THE LAST ROW NUMBER IN THE LOOK_UP_TABLE (i.e., 4)

        /*      Now, using the defined group values in the last row of the LOOK_UP array and
                the threshold for the error value, the rho values in the bin will be
                categorized. If any of the groups has more votes than threshold value, then
                this bin is marked for the second pass detection routine. */
```

P148371.PDF [Page: 19 of 61]


```
/*      LOOK_UP(4,0) contains the number of possible different groups in the 4th row.
        Check all the groups one by one. If there is at least one group
        (corresponding to a target) which satisfies the vote threshold then stop the
        process and mark the bin with a 1. (indicates the target presence) */

DO WHILE (I  <= LOOK_UP(INDEX,0))

        /*      Set the number of returns satisfying the above condition to 0 */

        SET COUNT TO 0

        /*      Go thru all the rho values in this bin of the histogram. If the
                return is in the group and the rho value of the return is less then
                the error threshold then increment the count by 1. */

        DO RET=1,OLD_RET(0,1)
            IF ((LOOK_UP(INDEX,I) = VECT(RET)) AND
                (ABS(HIST(RET)) < THRESHOLD_ERR)) THEN
                        COUNT = COUNT+1
            END IF
        END DO

        /*      If the number of votes for this group is greater then vote threshold,
                mark the bin and stop searching for other parallel targets. */

        IF (COUNT >= THRESHOLD_VOTE) THEN

                /*      Calculate the number of bins for the histogram corresponding
                        to this bin in the second phase detection and store the
                        initial angle (i.e., LOWER_LIMIT) and the number of bins in
                        the temporary array used by SECOND_PASS_DETECTION. */

                INCREMENT NR_OF_SEC_PASS_DATA BY 1
                IF (BIN = 1) THEN

                        /*      Use the theta value corresponding to the first bin
                                as lower limit for the second histogram */

                        SEC_PASS_DATA(NR_OF_SEC_PASS_DATA,1) = 0.5*DELTA_THETA1


                        /* Magnify two intervals (bins) */

                        SEC_PASS_DATA(NR_OF_SEC_PASS_DATA,2) =
                                2 * (DELTA_THETA1/DELTA_THETA2)

                ELSE
                        /*      Use the theta value for the previous bin as lower
                                limit  for the second histogram */

                        SEC_PASS_DATA(NR_OF_SEC_PASS_DATA,1) =
                                ((BIN-1)-0.5)*DELTA_THETA1

                        IF (BIN = 18) THEN

                                /*      Magnify two intervals (bins) */

                                SEC_PASS_DATA(NR_OF_SEC_PASS_DATA,2) =
                                        2 * (DELTA_THETA1/DELTA_THETA2)

                        ELSE
                                /*      Magnify three adjacent intervals (bins) */

                                SEC_PASS_DATA(NR_OF_SEC_PASS_DATA,2) =
                                        3 * (DEL_THETA1/DEL_THETA2)
                        END IF
                END IF
```

P148371.PDF [Page: 20 of 61]


```
                        I=LOOK_UP(INDEX,0)+1

                    END IF
                    I=I+1
                END DO

                /*      Clear the VECT array for the next bin process */

                DO I=1,OLD_RET(0,1)
                    VECT(I)=0
                END DO
            END DO
```

3.2.4 SECOND_PASS_DETECTION Routine

3.2.4.1 Description

This routine handles the second phase of the track initiation process. Appropriate routines are called to complete the track detection process.

SEC_PASS_DATA array is used to built the second phase histograms for the marked columns of the first phase histogram.

The BUILD_HISTOGRAM and the DETECT_TARGET routines are called to build the new histogram bins.

For the detected tracks (satisfying the vote_threshold) in the new histogram, NEW_TRACKS array is initialized by calling INITIATE_TRACK routine.

If a target is detected in one of the bins, before calling the initiation routine, this procedure sorts the plots in the bin by scan, and deletes the points which has the same time stamp occurred due to the clutter, chooses only one plot per scan. Duplications (in time and scan) are deleted on the basis of variation comparison.

This routine also sets the global variable PARAM(5) to number of the new tracks initiated.

This routine is only called by the INIT1 routine.

3.2.4.2 Inputs

REAL DEL_THETA1:
The parameter spacing value for the first phase of the detection (This value is chosen as 10 degrees)

P148371.PDF [Page: 21 of 61]

REAL DEL_THETA2:
The parameter spacing value for the second phase of the detection (This value is chosen as 1.0 degree)

REAL NEW_TRACKS(LIM_RET, 7):
where LIM_RET is a user defined parameter. This is a temporary array used to pass the information about the new tracks created by INIT1 routine to the ADD_TRACKS routine.

3.2.4.3 Outputs

REAL NEW_TRACKS(LIM_RET, 7):
See Inputs above.

REAL PARAM(5):
5th location of the PARAM array, which keeps the number of new tracks initiated in this scan.

3.2.4.4 Global Data

REAL PARAM(5):
5th location of the PARAM array, which keeps the number of new tracks initiated in this scan.

REAL PI:
literal value for the 180 degrees

INTEGER MAX_NR_OF_SCANS:
This variable keeps the total number of scans to be stored in order to invoke the initiation process.

REAL OLD_RET(0:LIM_RET*MAX_NR_OF_SCANS, 11):
This array keeps the returns from a number of previous scans. It has the same structure as RETURNS array explained in Sec 3.1.1 as A array.

REAL HIST(0:LIM_RET*MAX_NR_OF_SCANS):
The histogram bin, rows corresponding to the returns. This array (i.e. bin) keeps the rho values of each return for a particular theta value.

REAL VECT(LIM_RET*MAX_NR_OF_SCANS):
Used in grouping of the returns that belong to the parallel tracks. See Section 4.2.

P148371.PDF [Page: 22 of 61]

REAL LOOK_UP(4,0:8):
Used in grouping of the returns that belong to the parallel tracks. See Section 4.2.

REAL SECOND_PASS_DATA(18,2):
Keeps the data for the identified bins by the first pass detection routine

INTEGER NR_OF_SEC_PASS_DATA:
Keeps the number of identified bins in the first pass detection phase

3.2.4.5 **Local Data**

REAL TRACK_DATA(MAX_NR_OF_SCANS):
This array keeps the selected index values of the returns which will be used in the track initiation. The index value is the stored location of the return in the OLD_RET array.

REAL THRESHOLD_ERR, THRESHOLD_VOTE:
The threshold values used to determine the target presence (error threshold and vote threshold) in the second phase of the initiation.

INTEGER THRESHOLD_LEV:
Indicates the maximum order of averaging.

INTEGER NR_OF_BINS:
The nr of bins of the histogram to be built

REAL LOWER_LIMIT:
The initial angle theta, used in building the histogram

REAL ERROR:
keeps the variations from the average rho value.

REAL SUM:
used to calculate the MEAN_VAL

INTEGER COL, BIN, ROW, RET, J, I, INDEX:
Used for indexing into the arrays

INTEGER FOUND:
Used as a flag while selecting the returns to initiate a track.

P148371.PDF [Page: 23 of 61]

INTEGER COUNT:
Keeps the number of returns which belongs to a specific target

INTEGER NEW_NUM:
Number of the new tracks initiated in this scan
INTEGER NR_OF_PNTS:
Number of the returns after the variation test.

INTEGER SCAN_NUM:
variable used to keep the scan number of the return with the minimum deviation

3.2.4.6 Main Body of the Routine

```
/*      Following are the constant values used in the track detection; THRESHOLD_ERR is the
        threshold value for the error, THRESHOLD_VOTE is the number of 'votes' required for
        track detection process. Note that these values are different than the ones defined
        for the initial detection stage.*/

DEFINE THRESHOLD_ERR AS 1500
DEFINE THRESHOLD_VOTE AS 4 (or MAX_NR_OF_SCANS)

SET THRESHOLD_LEV TO 3 (highest order in averaging)

/* initialize the variable NEW_NUM, and NR_OF_SECOND_PASS_DATA*/
SET NEW_NUM TO 1
SET NR_OF_SECOND_PASS_DATA TO 0

/* Initialize the INDEX variable to the last row index of the LOOK_UP table. */

INDEX = THRESHOLD_LEV+1
DO COL=1, NR_OF_SEC_PASS_DATA
  LOWER_LIMIT = SEC_PASS_DATA(COL,1)
  NR_OF_BINS = SEC_PASS_DATA(COL,2)

/*  The histogram is built one bin at a time, and for each bin the presence of the target is
    checked. The detected targets are initialized in NEW_TRACKS array before the next bin in
    the histogram is built. */

  DO BIN=1 TO NR_OF_BINS
    CALL BUILD_HISTOGRAM(BIN, LOWER_LIMIT, DEL_THETA2)
    CALL DETECT_TARGET(BIN, THRESHOLD_LEV)

    /*  LOOK_UP(4,0) contains the number of possible different groups in the 4th row. Check
        all the groups one by one. When a group in the VECT array satisfies the vote threshold
        then the returns in that group with the smallest error values are selected to initiate
        the track. */

    DO J = 1 TO LOOK_UP(INDEX,0)

    /*  Set the number of returns satisfying the above condition to 0 */

      SET COUNT TO 0

      DO RET = 1 TO OLD_RET(0,1)

        /*  Go thru all the rho values in this bin of the histogram. If the return is in the
            group and the rho value of the return is less then the error threshold then if
            another return for the same scan is already stored in the TRACK_DATA array, check
```

P148371.PDF [Page: 24 of 61]


```
                the error values. Store the smallest. In this approach, only one (with the
                smallest error value) return  per scan will be stored in the TRACK_DATA array. */

              IF (OLD_RET(RET,10) = 0.0)
                /* If the return is not used before */

                IF ((LOOK_UP(INDEX,J) = VECT(RET)) AND (ABS(HIST(RET)) < THRESHOLD_ERR)) THEN

                  FOUND = 0
                  /*   If another return for the same scan is found to be stored in the TRACK_DATA
                       array then the FOUND flag is set to 1 */

                  /*   Go thru all the returns stored so far */

                  DO I=1 TO COUNT
                    SET ROW TO TRAK_DATA(I)
                    IF (OLD_RET(ROW,11) = OLD_RET(RET,11)) THEN
                       /*  Found another returns for the same scan */
                       IF (ABS(HIST(RET)) < ABS(HIST(ROW)) THEN
                         TRACK_DATA(I) = RET
                       END IF
                       FOUND = 1
                    END IF
                  END DO

                  /*   There are no returns stored for this scan store this one and increment the
                       count */

                  IF (FOUND = 0) THEN
                    COUNT = COUNT +1
                    TRACK_DATA(COUNT) = RET
                  END IF
                END IF
              END IF
            END DO

            /*  If the number of votes for one of the groups in the LOOK_UP table is greater than
                the preset threshold, then a new track is started with the two latest returns in
                that group */
            IF (COUNT > THRESHOLD_VOTE) THEN
              CALL INITIATE_TRACK(TRACK_DATA, COUNT, NEW_TRACKS, NEW_NUM)
            END IF
          END DO

          /*  Clear the VECT array for the next bin process */

          DO I=1,OLD_RET(0,1)
            VECT(I)=0
          END DO

        END DO
      END DO

      PARAM(5) = NEW_NUM - 1
```

3.2.5 BUILD_HISTOGRAM Routine Design

3.2.5.1 Description

This routine builds one histogram bin for a given DELTA_THETA and calculates the RHO values for all the returns in the OLD_RET table.

P148371.PDF [Page: 25 of 61]

The row index of the HIST array (i.e., bin) corresponds to the index into the returns in the OLD_RET table. The bin value represent the theta interval and is passed to this procedure by the calling routine. Also the start value for the theta (LOWER_LIMIT) is supplied by the calling routine.

This routine stores only the positive RHO values since negative values are invalid for the RHO. The VECT array is also initialized for the positive RHO values with a one.

This routine is called by INITIAL_DETECTION, SECOND_PASS_DETECTION.

3.2.5.2 Inputs

REAL DELTA_THETA:
The parameter spacing value used for building the histogram

INTEGER NR_OF_BINS:
The number of bins of the histogram

REAL LOWER_LIMIT:
Initial angle (theta value) to start building the histogram.

3.2.5.3 Outputs

REAL HIST(0:LIM_RET*MAX_NR_OF_SCANS):
The histogram array, rows corresponding to the returns and the columns corresponding to the bins of the histogram. This array keeps the rho values for each return with different theta value

3.2.5.4 Global Data

INTEGER LIM_RET:
The number of maximum returns per scan that ATP can process

INTEGER MAX_NR_OF_SCANS:
This variable keeps the literal value for total number of scans to be stored in order to invoke the initiation process.

P148371.PDF [Page: 26 of 61]

REAL OLD_RET(0:MAX_NR_OF_SCANS* LIM_RET, 11):
This array keeps the return from a number of previous scans. It has the same structure as RETURNS array explained in Section 3.1.1 as A array.

REAL HIST(0:MAX_NR_OF_SCANS*LIM_RET):
The histogram array, rows corresponding to the returns and the columns corresponding to the bins of the histogram. This array keeps the rho values for each return with different theta value

INTEGER VECT(4,0:8):
Used in grouping returns corresponding to different parallel tracks. See Section 4.2.

3.2.5.5 Local Data

REAL X, Y:
X and Y position of the return (i.e, RETURN(x,1), RETURN (x, 2))

REAL THETA:
The calculated angle for each bin using the given parameter spacing (DELTA_THETA)

INTEGER ROW:
Used for indexing into the HIST and OLD_RET arrays.

3.2.5.6 Main Body of the Routine

```
CALCULATE THETA AS:  LOWER_LIMIT+( (BIN-0.5)*DELTA_THETA )

DO ROW=1,OLD_RET(0,1)

    IF (OLD_RET(ROW,10) = 0.0 ) THEN

         /* If the return is valid */

         SET X TO OLD_RET(ROW,1)
         SET Y TO OLD_RET(ROW,2)

         CALCULATE RHO AS (X*COS(THETA)+Y*SIN(THETA))

         /*  RHO indicates the distance. Therefore only the positive values are valid.
             Therefore, store the RHO value in the bin only if RHO is positive.
             The VECT array is also initialized according to the sign of the RHO value */
         IF (RHO > 0) THEN
           SET HIST(ROW) TO RHO
           SET VECT(ROW) TO 1
         ELSE
           SET VECT(ROW) TO 0
         END IF
    ENDIF
END DO
```

P148371.PDF [Page: 27 of 61]

3.2.6 DETECT_TARGET Routine Design

3.2.6.1 Description

This routine goes through the given bin of the histogram and categorizes the rho values to the groups which correspond to the potential parallel track candidates. The groups are formed according to the sign of the distances from the average rho value (ERROR). This grouping process is carried on for three times.

Technique:

Let say there are 7 returns (7 positive rho values); R1, R2, R3, R4, R5, R6 and R7.

First Time Averaging

Mean Value for group 1	VECT Array
R6	
R3	VECT(1) = 11
R4	VECT(2) = 11
R2	VECT(3) = 11
R1	VECT(4) = 11
---------- Mean Rho Value	VECT(5) = 12
R5	VECT(6) = 11
R7	VECT(7) = 12

Second Time Averaging

Mean Value for group 11	VECT Array
R6	
R3	VECT(1) = 112
R4	VECT(2) = 111
R2	VECT(3) = 111
----------Mean Value for group 11	VECT(4) = 111
R1	VECT(6) = 111

P148371.PDF [Page: 28 of 61]

Mean Value for group 12 VECT Array

R7
----------Mean Value for group 12 VECT(5) = 122
R5 VECT(7) = 121

Third Time Averaging

Mean Value for group 111 VECT Array

R3 VECT(2) = 1111
R4 VECT(3) = 1111
R2 VECT(4) = 1111
----------Mean Value for group 111 VECT(6) = 1112
R6

Only Group 111 satisfies the vote threshold, (i.e., three returns found in the group), therefore a track will be started using the most recent two returns amongst the returns R2, R3 and R4.

This routine is called by the routines INITIAL_DETECTION and SECOND_PASS_DETECTION.

3.2.6.2 **Inputs**

REAL THRESHOLD_LEV:
The threshold value used to determine the maximum number of order in averaging technique.

INTEGER BIN:
The number of the histogram bin to be examined.

3.2.6.3 **Outputs**

REAL HIST(0:MAX_NR_OF_SCANS*LIM_RET):
The array used to represent the two dimensional histogram. If a target presence is detected in a bin then the 0th location of the corresponding bin marked with a 1, 0 otherwise.

P148371.PDF [Page: 29 of 61]

3.2.6.4 **Global Data**

REAL HIST(0:MAX_NR_OF_SCANS*LIM_RET):
This array is used to represent the two dimensional histogram. If a target presence is detected in a bin then the 0th location of the corresponding bin marked with a 1, 0 otherwise.

REAL OLD_RET(0:MAX_NUMBER_OF_SCANS*LIM_RET, 11):
This array keeps the returns from a number (MAX_NR_OF_SCANS) of previous scans. It has the same structure as RETURNS array explained in Section 3.1.1 as A array.

INTEGER MAX_NR_OF_SCANS:
The literal value for the duration of the waiting period in scans

INTEGER VECT(LIM_RET*LIM_RET):
Used in grouping returns corresponding to different parallel tracks. See Section 4.2.

INTEGER LOOK_UP(4,0:8):
Used in grouping returns corresponding to different parallel tracks. See Section 4.2.

3.2.6.5 **Local Data**

REAL SUM:
the total sum of the rho values in a histogram bin

REAL ERROR:
the error value between the rho value and the mean value of a bin

REAL MEAN:
The calculated average rho value

INTEGER NR_OF_DATA:
number of radar data in a bin to be used in the track initiation

INTEGER RET, J, I:
used for indexing into the arrays

P148371.PDF [Page: 30 of 61]

3.2.6.6 Main Body of the Routine

```
DO I=1 TO THRESHOLD_LEV

  DO J=1 TO LOOK_UP(I,0)

    SET SUM TO 0.0
    SET NR_OF_DATA TO 0

    /*  Go thru each all the rho values in the bin corresponding the returns*/

    DO RET=1 TO OLD_RET(0,1)

      /*  If the return is not used  before */
      IF (OLD_RET(RET,10) = 0.0) THEN

           /*    Sum up all the RHO values in the bin and calculate the number of rho
                values */

           IF (LOOK_UP(I,J) = VECT(RET)) THEN
             SUM=SUM+HIST(RET)
             NR_OF_DATA=NR_OF_DATA+1
           END IF
      END IF
    END DO

    /*  If there are any returns corresponding to this group, calculate the average value,
        and mark the groups for each return in the VECT array. */

    IF (NR_OF_DATA > 0 ) THEN
      MEAN = SUM/NR_OF_DATA

      DO RET=1, OLD_RET(0,1)
        IF (OLD_RET(RET,10) = 0.0) THEN
          IF (LOOK_UP(I,J) = VECT(RET)) THEN
            ERROR = HIST(RET)-MEAN
            HIST(RET) = ERROR

            IF (ERROR < 0.0) THEN

              /*  Rho is below the average */
              VECT(RET) = VECT(RET)*10+2

            ELSE
              /*  Rho is below the average */
              VECT(RET) = VECT(RET)*10+1
            END IF
          END IF
        END IF
      END DO
    END IF
  END DO
END DO
```

3.2.7 INITIATE_TRACK Routine Design

3.2.7.1 Description

This routine initiates potential tracks in the NEW_TRACKS array by using the data in the TRACK_DATA array.

P148371.PDF [Page: 31 of 61]

The NEW_TRACKS array is initialized by using the two latest returns (on the straight line). The validity field of the OLD_RET array corresponding to the data used for the track initiation, is set to -999.99 to prevent the reuse of same radar data.

This routine also applies a speed threshold test before it initiates the track.

This routine is only called by the SECOND_PASS_DETECTION routine.

3.2.7.2 Inputs

REAL NEW_TRACKS(LIM_RET, 7):
where LIM_RET is a user defined parameter. This is a temporary array used to pass the information about the new tracks to the ADD_TRACKS routine.

INTEGER NEW_NUM:
The number of new tracks initiated in this scan

REAL TRACK_DATA(MAX_NR_OF_SCANS):
This array keeps indices into OLD_RET array for the returns which will be used to initiate a track.

INTEGER NR_OF_PNTS:
size of TRACK_DATA array.

3.2.7.3 Outputs

REAL NEW_TRACKS(LIM_RET, 7): See Inputs above

INTEGER NEW_NUM: See Inputs above

3.2.7.4 Global Data

REAL OLD_RET(MAX_NR_OF_SCANS*LIM_RET, 11):
This array keeps the return from a number of previous scans. It has the same structure as RETURNS array explained in Sec. 3.1.1 as A array.
REAL PARAM(6), PARAM(7): Maximum and minimum speed of the target defined by user in Case.dat

P148371.PDF [Page: 32 of 61]

3.2.7.5 Local Data

REAL SCAN_NR:
scan id of the most recent return

REAL Vx, Vy:
calculated velocity of the track in x and y direction respectively.

REAL V:
calculated range speed

REAL V_MIN, V_MAX:
Maximum and Minimum speed of the target

REAL X1, X2, Y1, Y2 :
variables used to store the calculated cartesian coordinates for the two most recent returns

REAL DELTA_T:
time difference between the two most recent returns

INTEGER INDEX1, INDEX2:
the index into the OLD_RET array for the most recent two returns in time.

INTEGER I, ROW:
used for indexing into the OLD_RET and TRACK_DATA arrays.

3.2.7.6 Main Body of the Routine

```
/* Initialize the minimum and maximum speed of the target */

SET V_MAX TO PARAM(6)
SET V_MIN TO PARAM(7)

/* Get the latest return in time */

SCAN_NR = -1
DO I=1 TO NR_OF_PNTS

  ROW = TRACK_DATA(I,1)

  IF (OLD_RET(ROW,11) > SCAN_NR) THEN
    SET SCAN_NR TO OLD_RET(ROW,11)
    SET INDEX1 TO ROW
  END IF
END DO

X1 = OLD_RET(INDEX1,1)
Y1 = OLD_RET(INDEX1,2)

/* Get the second latest return in time */
```

P148371.PDF [Page: 33 of 61]


```
SCAN_NR = -1
DO I=1 TO NR_OF_PNTS
  ROW = TRACK_DATA(I,1)

  IF (OLD_RET(ROW,11) > SCAN_NR) THEN
    IF (OLD_RET(ROW,11) <> OLD_RET(INDEX1,11)) THEN
      SET SCAN_NR TO OLD_RET(ROW,11)
      SET INDEX2 TO ROW
    END IF
  END IF

END DO

X2 = OLD_RET(INDEX2,1)
Y2 = OLD_RET(INDEX2,2)

/*  Check the speed thresholds before initiating the track */

DELTA_T = OLD_RET(INDEX1,3)-OLD_RET(INDEX2-3)
VX=(X1-X2)/DELTA_T
VY=(Y1-Y2)/DELTA_T
V=SQRT(VX*VX + VY*VY)

IF ((V_MIN <= V) AND (V <= V_MAX)) THEN

  /*  Initialize the NEW_TRACK array with latest data on the calculated line */

  SET NEW_TRACKS(NEW_NUM,1) TO X1
  SET NEW_TRACKS(NEW_NUM,2) TO Y1

  SET NEW_TRACKS(NEW_NUM,3) TO Vx
  SET NEW_TRACKS(NEW_NUM,4) TO Vy

  SET NEW_TRACKS(NEW_NUM,5) TO OLD_RET(INDEX1,3)
  SET NEW_TRACKS(NEW_NUM,6) TO OLD_RET(INDEX1,4)
  SET NEW_TRACKS(NEW_NUM,7) TO OLD_RET(INDEX1,5)
  OLD_RET(INDEX1,10) = -999.99
  NEW_NUM=NEW_NUM+1
END IF
```

3.3 Logic-Based Track Initiation

The Logic-Based Track Initiation Technique is discussed in Section 2.2.3 of the Software Specification Document for Task I of the MTT Phase II project. The inputs, outputs and the algorithm are also described in Section 2.2.3 in a general sense.

Following sections document the detailed design of the routines to be implemented for the Logic-Based Track Initiation Technique.

3.3.1 INIT1 Routine Design

3.3.1.1 Description

This is the control routine for the track initiation process. It calls the appropriate routines to carry out the Logic-Based track initiation.

P148371.PDF [Page: 34 of 61]

First it calls the association routine and the track update routine to associate the current returns with the existing potential tracks and to update the potential tracks with the associated returns. The returns used in the track update process are discarded from further consideration.

INIT1 routine, then calls the association process again to associate the returns to the returns from the previous scan. The results of the association process are passed to the track initiation process. The associated returns are used to initiate the new potential tracks and are marked as used.

Finally, the SAVE_CONTACT routine is called to save the unused returns from the current scan into the OLD_RET array, for the next scan track initiations.

This routine is only called by FRAME routine.

3.3.1.2 **Inputs**

REAL RETURNS(0:LIM_RET, 15):
This array contains the information for all the returns from the current scan. The RETURNS array is explained in Sec. 3.1.1 as A array.

REAL NEW_TRACKS(LIM_RET,7):
Where LIM_RET is user-defined parameter. This is a temporary array used to pass the information about the new tracks initiated by INIT1 routine to the ADD_TRACKS routine.

3.3.1.3 **Outputs**

REAL NEW_TRACKS(LIM_RET,7):
See Inputs above

3.3.1.4 **Global Data**

REAL POT_TRACKS(0:N*LIM_RET,40):
N will be defined in a later stage (changes with the density of the clutter). This array keeps information about the potential tracks and is explained in Sec. 4.3 in detail.

REAL OLD_RET(0:LIM_RET,11):
This array keeps the returns from the previous scans. The location (0,1) keeps

P148371.PDF [Page: 35 of 61]

the number of the old returns. It has the same structure as the RETURNS array explained in Section 3.1.1 as A array.

INTEGER CONTACT_ASSOC:
This variable is used as a flag for the association process. It indicates the association process between returns from the current scan and the returns from the previous scan. Initialized to 1.

INTEGER TRACK_ASSOC:
This variable is used as a flag for the association process. It indicates the association process between returns from the current scan and the returns from the previous scan. Initialized to 2.

INTEGER LIM_RET:
The maximum number of returns per scan that is processed by ATP routine.

REAL ASSOC_ARRAY(LIM_RET):
This array is used to store the results of the association process. Association process associates either

- current returns to old returns or;
- current returns to potential tracks.

The indexing of the ASSOC_ARRAY is the same as RETURNS array indexing. (i.e., ASSOC_ARRAY(1) keeps information about the first element in RETURNS array).

In the case of the return/old_return association process, this array keeps the index values of the associated old returns in the OLD_RET table.

In the case of the association process of the potential tracks/returns, it keeps the index values of the associated potential tracks in the POT_TRACKS array.

If any of the locations in the ASSOC_ARRAY contains a -1 value, this signifies that corresponding current return is not associated.

INTEGER COUNT:
This global variable is set to 0 after every initiation process. This variable is used to store the number of the processed scans during initialization.

INTEGER MAX_NR_OF_SCANS:
This variable keeps the maximum number of scans to be processed by initiation routine before the track confirmation.

P148371.PDF [Page: 36 of 61]

REAL PARAM(5):
5th location of the PARAM array, which keeps the number of new tracks initiated by INIT1.

3.3.1.5 Local Data

INTEGER NEW_NUM:
This variable keeps the number of the new tracks created by the track initiation process (INIT1).

INTEGER TRK_ID:
used for indexing into the POT_TRK array

REAL DELTA_X, DELTA_Y, DELTA_T:
Temporary variables used in the calculation of the target velocity

3.3.1.6 Main Body of the Routine:

```
IF (COUNT <= MAX_NR_OF_SCANS) THEN

  PARAM(5) = 0
  INCREMENT COUNT BY 1

  /*  If any potential track exists then try to associate the returns with the potential
      tracks first.*/

  IF (POT_TRK(0,1) > 0.0) THEN

    /*  Perform the potential-track/plot association for the existing potential tracks and
        update the tracks, the plots associated with the existing tracks will be removed
        from consideration. The results of the association process will be stored in the
        ASSOC_ARRAY. The index into the ASSOC_ARRAY corresponds to the one into the
        RETURNS array. ASSOC_ARRAY keeps the index numbers into the POT_TRACKS array for
        the associated potential tracks. */

    CALL ASSOCIATION_PROC(RETURNS,TRACK_ASSOC)

    /*  Update the potential tracks with the associated plot from this scan */

    CALL UPDATE_POT_TRACKS(RETURNS)

  END IF

  /*  If the OLD_RET array is not empty, perform potential track initiation. (Note that
      OLD_RET(0,1) keeps the number of old returns saved in the OLD_RET array) */

  IF (OLD_RET(0,1) > 0.0) THEN

    /*  For the plots which are not associated with any of the tracks and not used in the
        potential track update, try to associate with the saved plots (OLD_RET) from the
        previous scans. The plots which are associated with the saved plots will be
        removed from consideration. The result of the association process will be stored
        in the ASSOC_ARRAY. The index into the ASSOC_ARRAY corresponds to the one into the
        RETURNS array. ASSOC_ARRAY keeps the index numbers into the OLD_RET array for the
        associated saved plots. */
```

P148371.PDF [Page: 37 of 61]


```
            CALL ASSOCIATION_PROC(RETURNS,CONTACT_ASSOC)

            /*  For the associated saved contact/plot pairs initiate the track */

            CALL INITIATE_POT_TRACKS(RETURNS)

          END IF

          /*  The unused plots will be saved in the OLD_RET table for the next scan track
              initiation process.*/

          SAVE_CONTACT(RETURNS)

        /*  If the nr of scans (COUNT) is greater than the maximum scan number, start the tracks
            in the global track database and clear the potential track table. */

        ELSE

          /*  Set the index to the NEW_TRACKS array to 1 and clear the COUNT variable for the next
              initiation period. */

          NEW_NUM = 1
          COUNT = 0

          DO TRK_ID = 1 TO POT_TRK(0,1)

            /* DELTA_T keeps the time difference between the latest two measurements */

            DELTA_T = POT_TRK(TRK_ID,35) -  POT_TRK(TRK_ID,30)

            /*  DELTA_X keeps the positional difference in x-direction */
            /*  DELTA_Y keeps the positional difference in y-direction */

            DELTA_X = POT_TRK(TRK_ID,36) - POT_TRK(TRK_ID,31)
            DELTA_Y = POT_TRK(TRK_ID,37) - POT_TRK(TRK_ID,32)

            NEW_TRACKS(NEW_NUM,1) = POT_TRK(TRK_ID,36)
            NEW_TRACKS(NEW_NUM,2) = POT_TRK(TRK_ID,37)
            NEW_TRACKS(NEW_NUM,3) = DELTA_X/DELTA_T
            NEW_TRACKS(NEW_NUM,4) = DELTA_Y/DELTA_T
            NEW_TRACKS(NEW_NUM,5) = POT_TRK(TRK_ID,35)
            NEW_TRACKS(NEW_NUM,6) = POT_TRK(TRK_ID,33)
            NEW_TRACKS(NEW_NUM,7) = POT_TRK(TRK_ID,34)

            INCREMENT THE NEW_NUM BY 1

            /* Set the POT_TRK validity field to invalid */
            POT_TRK(TRK_ID,1) = 0.0

          END DO
        END IF

        /*  Clear the number of potential tracks,and store the number of tracks created this scan
            in PARAM(5) */

        POT_TRK(0,1) = 0.0
        PARAM(5) = NEW_NUM-1
      END IF
```

P148371.PDF [Page: 38 of 61]

3.3.2 INITIATE_POT_TRACKS Routine Design

3.3.2.1 Description

This module initiates potential tracks using the results of the association process. If the return is associated with any of the returns from the previous scan, then the ASSOC_ARRAY contains the index of the old return stored in the OLD_RET table. In this case the data for the first set of measurement is retrieved from the OLD_RET array by using the index value in ASSOC_ARRAY.

This module calculates the covariance and position estimates for the potential track, searches for an available slot in the POT_TRACKS array and stores the necessary tracking values in the POT_TRACKS. POT_TRACKS array is explained in section 4.3 in detail.

This routine is called by INIT1 module only.

3.3.2.2 Inputs

REAL RETURNS(0:LIM_RET, 15):
This array contains the information for all the returns from the current scan. The RETURNS array is explained in Sec. 3.1.1 as A array.

3.3.2.3 Outputs

REAL POT_TRACKS(N*LIM_RET,40):
This table keeps the information about the potential tracks created by the initiation process. POT_TRACKS array is described in section 4.3 in detail.

3.3.2.4 Global Data

REAL POT_TRACKS(N*LIM_RET,40):
This table keeps the information about the potential tracks created by the initiation process. POT_TRACKS array is described in section 4.3 in detail.

REAL PARAM(11), PARAM(12):
This routine accesses the 11th and the 12th locations of the PARAM array to retrieve the range and the bearing variances of the measurement respectively.

P148371.PDF [Page: 39 of 61]

INTEGER LIM_RET:
The maximum number of returns per scan that is processed by ATP routine.

REAL ASSOC_ARRAY(LIM_RET):
This array is used to store the results of the association process. Association process associates either

- current returns to old returns or;
- current returns to potential tracks.

The indexing of the ASSOC_ARRAY is the same as RETURNS array indexing. (i.e., ASSOC_ARRAY(1) keeps information about the first element in RETURNS array).

In the case of the return/old_return association process, this array keeps the index values of the associated old returns in the OLD_RET table.

If any of the locations in the ASSOC_ARRAY contains a -1 value, this signifies that corresponding current return is not associated.

3.3.2.5 Local Data

REAL R_VEL_CAL, B_VEL_CAL:
These variables are used to store the calculated velocities of the target.

REAL V_MIN(2), V_MAX(2):
These arrays keep the minimum and the maximum speed thresholds for range and bearing

REAL SIGMA(2):
contains the measurement variances for the range and the bearing.

REAL COEF(2):
contains the squared value of the measurement variances.

REAL T_1:
contains the time of the first measurement.

REAL T_2:
contains the time of the second measurement.

REAL M_1(2):
contains the range and bearing values for the first set of measurements.

P148371.PDF [Page: 40 of 61]

REAL M_2(2):
contains the range and bearing values for the second set of measurements.

REAL T:
used to store the time difference between the first and the second measurement divided by 2.

INTEGER CONTACT:
used as the index into the RETURN array.

TEGER OLD_CONTACT:
used as the index into the OLD_RET array

REAL A0,A1:
Coefficients of the state equation

INTEGER CREATE_FLAG:
used as a flag variable to control the logic flow

The variables I, J, K, INDEX, TRK_IDX are used in loop and index control.

3.3.2.6 Main Body of the Routine

```
INITIALIZE K TO 2

/*  SIGMA(1) = variance of radar range measurement
  SIGMA(2) = variance of radar bearing measurement*/

SET SIGMA(1) TO SQRT(PARAM(11))
SET SIGMA(2) TO SQRT(PARAM(12))

DO I=1 TO 2
  COEF(I) SIGMA(I)*SIGMA(I)/2
END DO

/* Set the minimum and maximum velocities */

V_MIN(1) = PARAM(7)
V_MAX(1) = PARAM(6)
V_MIN(2) = PARAM(7)/150000 ! 150000 is the maximum detection range
V_MAX(2) = PARAM(6)/6000 ! 6000 is the minimum detection range

/* Note that RETURNS(0,1) keeps the number of returns for this scan */

DO CONTACT=1 TO RETURNS(0,1)

/* If we are processing the general search initiation get the range and bearing values of
the measurements and the delta_t accordingly */

  IF (ASSOC_ARRAY(I) <> -1) THEN

    OLD_CONTACT = ASSOC_ARRAY(CONTACT)
    T = (RETURNS(CONTACT,3) - OLD_RET(OLD_CONTACT,3))/2
    T_1 = OLD_RET(OLD_CONTACT,3)
```

P148371.PDF [Page: 41 of 61]


```
            T_2 = RETURN(CONTACT,3)

            M_1(1) = OLD_RET(OLD_CONTACT,4)
            M_1(2) = OLD_RET(OLD_CONTACT,5)

            M_2(1) = RETURNS(CONTACT,4)
            M_2(2) = RETURNS(CONTACT,5)

            R_VEL_CAL = ABS ( (M_2(1)-M_1(1)) / (2*T) )
            B_VEL_CAL = ABS ( (M_2(2)-M_1(2)) / (2*T) )

            /* Check the speed thresholds */
            IF ( (R_VEL_CAL >= V_MIN(1) ) AND
                 (R_VEL_CAL <= V_MAX(1) ) AND
                 (B_VEL_CAL >= V_MIN(2) ) AND
                 (B_VEL_CAL <= V_MAX(2) ) ) THEN

                 CREATE_FLAG = 1

            END IF

        END IF

        IF (CREATE_FLAG > 0) THEN

            /* Write the potential track in the table */

            TRK_IDX = POT_TRK(0,1)+1

            POT_TRK(TRK_IDX,1) = 1 ! Valid flag
            POT_TRK(TRK_IDX,3) = RETURNS(CONTACT,3) !TIME

            /* POT_TRACKS array location 4-12 contain covariance estimate for range */
            /* POT_TRACKS array location 13-21 contain covariance estimate for bearing */

            /* Clear the table before writing the track */

            DO INDEX = 4 TO 27
              POT_TRK(TRK_IDX,INDEX) = 0.0
            END DO

            POT_TRK(TRK_IDX,4) = COEF(1)
            POT_TRK(TRK_IDX,13) = COEF(2)
            POT_TRK(TRK_IDX,8) = 2*COEF(1)/(4*T*T)
            POT_TRK(TRK_IDX,17) = 2*COEF(2)/(4*T*T)

            /* POT_TRACKS array location 22-24 contain position vector for range */
            /* POT_TRACKS array location 25-27 contain position vector for bearing */
            INDEX = 22
            DO I=1 TO 2
              A0 = (M_1(I) + M_2(I))/2
              A1 = (M_2(I) - M_1(I))/(2*T)
              POT_TRK(TRK_IDX,INDEX) = A0+A1*T
              INDEX = INDEX +1
              POT_TRK(TRK_IDX,INDEX) = A1
              INDEX = INDEX+2
            END DO

            /* POT_TRACKS array location 28-32 contain Range, Bearing, Time, X-pos and Y-pos for
               the measurement 1 */

            POT_TRACKS(TRK_IDX,28) = M_1(1)
            POT_TRACKS(TRK_IDX,29) = M_1(2)
            POT_TRACKS(TRK_IDX,30) = T_1
            POT_TRACKS(TRK_IDX,31) = OLD_RET(OLD_CONTACT,1)
            POT_TRACKS(TRK_IDX,32) = OLD_RET(OLD_CONTACT,2)
```

P148371.PDF [Page: 42 of 61]


```
            /* POT_TRACKS array location 33-37 contain Range, Bearing, Time, X-pos and Y-pos for
               the measurement 2 */

            POT_TRACKS(TRK_IDX,33) = M_2(1)
            POT_TRACKS(TRK_IDX,34) = M_2(2)
            POT_TRACKS(TRK_IDX,35) = T_2
            POT_TRACKS(TRK_IDX,36) = RETURNS(CONTACT,1)
            POT_TRACKS(TRK_IDX,37) = RETURNS(CONTACT,2)

            /* POT_TRACKS array location 38 contains update flag (0 if the potential track is not
               updated this scan*/

            POT_TRACKS(TRK_IDX,38) = 1.0

            /* location 39 is empty */

            /* Increment the number of the potential tracks */

            POT_TRK(0,1) = POT_TRK(0,1)+ 1

            /* Remove the contact from further consideration */

            RETURNS(CONTACT,10) = -999.99

          END IF
        END IF
      END DO
```

3.3.3 SAVE_CONTACT Routine Design

3.3.3.1 Description

SAVE_CONTACT routine first clears out the OLD_RET array. Then it saves the unused returns from the current scan into the OLD_RET array for the next initiation cycle.

This routine is called by INIT1 module only.

3.3.3.2 Inputs

REAL RETURNS(0:LIM_RET, 15):
This array contains the information for all the returns from the current scan. The RETURNS array is explained in Sec. 3.1.1 as A array.

3.3.3.3 Outputs

REAL OLD_RET(0:LIM_RET,11):
This array keeps the returns from the previous scans. The location (0,1) keeps the number of the old_returns. It has the same structure as A array described in Sec. 3.1.1.

P148371.PDF [Page: 43 of 61]

3.3.3.4 Global Data

REAL OLD_RET(0:LIM_RET,11):
This array keeps the returns from the previous scans. The location (0,1) keeps the number of the old_returns. It has the same structure as A array described in Sec. 3.1.1.

LIM_RET:
The maximum number of returns per scan that is processed by ATP routine.

3.3.3.5 Local Data

INTEGER I, J, INDEX:
These variables are used for loop and array indexing.

3.3.3.6 Main Body of the Routine

```
/*  First remove all the contacts recorded previous scan*/

DO I=1 TO OLD_RET(0,1)
  OLD_RET(I,10) = -999.99
END DO

INDEX = 1

DO I=1 TO RETURNS(0,1)

  IF (RETURNS(I,10) - 0.0)

    DO J=1 TO 11
      OLD_RET(INDEX,J) = RETURNS(I,J)
    END DO

    INDEX = INDEX+1

  END IF

END DO

OLD_RET(0,1) = INDEX - 1
```

3.3.4 ASSOCIATION_PROC Routine Design

3.3.4.1 Description

This routine associates the current returns with either the old returns or the potential tracks, depending on the value of the FLAG variable. If we are associating returns with the old returns, the window is built around the old return measurements. If the returns are associated with the potential tracks

P148371.PDF [Page: 44 of 61]

then the window is built around the predicted position of the potential tracks. For the returns which fall into the window, the statistical distances are calculated. The candidate with a minimum statistical distance is associated with the old_return or the potential track, and the index of the candidate (either into the POT_TRACKS array or the OLD_RET array) is entered in the ASSOC_ARRAY.

This routine is called by INIT1 module only.

3.3.4.2 Inputs

REAL RETURNS(0:LIM_RET, 15):
This array contains the information for all the returns from the current scan. The RETURNS array is explained in Sec. 3.1.1 as A array.

INTEGER FLAG:
Indicates the type of association. If this variable is equal to CONTACT_ASSOC (literally 1) then returns are associated to old returns. If it is equal to TRACK_ASSOC (literally 2) then returns are associated to the Potential Tracks.

3.3.4.3 Outputs

REAL ASSOC_ARRAY(LIM_RET):
This array is used to store the results of the association process. Association process associates either

- current returns to old returns or;
- current returns to potential tracks.

The indexing of the ASSOC_ARRAY is the same as RETURNS array indexing. (i.e., ASSOC_ARRAY(1) keeps information about the first element in RETURNS array).

In the case of the return/old_return association process, this array keeps the index values of the associated old returns in the OLD_RET table.

In the case of the association process of the potential tracks/returns, it keeps the index values of the associated potential tracks in the POT_TRACKS array.

P148371.PDF [Page: 45 of 61]

3.3.4.4 **Global Data**

REAL POT_TRACKS(N*LIM_RET,40):
This table keeps the information about the potential tracks created by the initiation process. POT_TRACKS array is described in section 4.3 in detail.

REAL PARAM(11), PARAM(12):
This routine accesses the 11th and the 12th locations of the PARAM array to retrieve the range and the bearing variances of the measurement respectively.

REAL OLD_RET(0:LIM_RET,11):
This array keeps the returns from the previous scans. The location (0,1) keeps the number of the old_returns. It has the same structure as the A array described in Sec. 3.1.1.

INTEGER CONTACT_ASSOC:
This variable is used as a literal to indicate the association between returns from the current scan and the returns from the previous scan. Initialized to 1.

INTEGER TRACK_ASSOC:
This variable is used as a literal to indicate the association between returns from the current scan and the returns from the previous scan. Initialized to 2.

LIM_RET:
The maximum number of returns per scan that is processed by ATP routine.

REAL ASSOC_ARRAY(LIM_RET):
This array contains the indexes into the OLD_RET array for the returns associated with any of the previous returns. This return is filled out by the ASSOC_PROCESS routine. The indexing of the ASSOC_ARRAY corresponds to the RETURNS array indexing. (eg., if ASSOC_ARRAY(1) contains 2, this means that RETURNS(1) is associated with OLD_RET(2)).

If any location of the ASSOC_ARRAY contains a -1, this means that the corresponding return is not associated with any of the previous returns stored in OLD_RET table.

3.3.4.5 **Local Data**

REAL MIN_Dij:
keeps the minimum statistical distance value

P148371.PDF [Page: 46 of 61]

REAL POS(2):
This array keeps the range and bearing values to build the window around

REAL POS_RET(2):
This array keeps the range and bearing values of the current contact

REAL MAX_VALUE(2):
keeps the value of the window size when only the max velocities are used

REAL MIN_VALUE(2):
keeps the value of the window size when only the min velocities are used

REAL Dij:
keeps the calculated statistical distance value

REAL dij(2):
keeps the calculated statistical distance components for the range and the bearing

REAL R_VEL_CAL, B_VEL_CAL:
These variables are used to store the calculated velocities of the target.

REAL V_MIN(2), V_MAX(2):
These arrays keep the minimum and the maximum speed thresholds for range and bearing

REAL R(2):
keeps the covariances for range and bearing for the statistical distance calculation.

REAL LAMBDA:
preset threshold value, (chi-square tables) chosen as 5.3

REAL T:
Time difference between the two sets of measurements

REAL P(21):
temporary storage for the values in locations 4 to 21 of POT_TRACKS array.

INTEGER I, J, IDX, CONTACT, LIMIT:
These variables are used for loop and array indexing.

INTEGER UPDATE_FLAG:
used for the flow control

P148371.PDF [Page: 47 of 61]

3.3.4.6 Main Body of the Routine

```
/* set the minimum and the maximum speeds */

V_MIN(1) = PARAM(7)
V_MAX(1) = PARAM(6)
V_MIN(2) = PARAM(7)/150000 ! 150000 is the maximum detection range
V_MAX(2) = PARAM(6)/6000 ! 6000 is the minimum detection range


/*  Initialize the association matrix */

DO I=1 TO LIM_RET
  ASSOC_ARRAY(I) = -1
END DO

LAMBDA = 5.3

/*  Work through each element of RETURNS table , Note that RETURNS(0,1) contains the
    number of current returns*/

DO CONTACT= 1 TO RETURNS(0,1)

  /*  If the return is not used the existing track update then proceed with the
      association */

  IF (RETURNS(CONTACT,10) = 0.0 )  THEN

    MIN_Dij = -1.0
    POS_RET(1) = RETURNS(CONTACT,4)
    POS_RET(2) = RETURNS(CONTACT,5)

      /*  Work through each element of OLD_RET table if we are associating the returns
          with the returns from the previous scans (potential track initiation). Otherwise
          work through each element of POT_TRACKS table (potential track update)*/

    IF (FLAG = CONTACT_ASSOC) THEN
      LIMIT = OLD_RET(0,1)
    ELSE
      LIMIT = POT_TRACKS(0,1)
    END IF

    DO IDX=1 TO LIMIT

      UPDATE_FLAG = 0
      IF (FLAG = CONTACT_ASSOC) THEN

        /*   If we are performing association between new returns and the returns from the
             previous scans get the positions from the previous scan from the OLD_RET
             array and calculate delta accordingly */
        /*   If we are associating the returns with the existing potential tracks, get the
             covariances for the corresponding track */

        T = RETURNS(CONTACT,3) - OLD_RET(IDX,3)
        POS(1) = OLD_RET(IDX,4)
        POS(2) = OLD_RET(IDX,5)
        R(1) = SQRT ( PARAM(11) )
        R(2) = SQRT ( PARAM(12) )

        R_VEL_CAL = ABS ( (POS_RET(1) - POS(1) ) / T )
        B_VEL_CAL = ABS ( (POS_RET(2) - POS(2) ) / T )

        /* Check the speed thresholds */
        IF ( (R_VEL_CAL >= V_MIN(1) ) AND
          (R_VEL_CAL <= V_MAX(1) ) AND
          (B_VEL_CAL >= V_MIN(2) ) AND
          (B_VEL_CAL <= V_MAX(2) ) ) THEN
```

P148371.PDF [Page: 48 of 61]


```
                UPDATE_FLAG = 1

              END IF
            ELSE

              IF (POT_TRK(IDX,1) = 1.0) THEN
                /*  If we are performing association between new returns and the existing
                    potential tracks calculate the predicted range and bearing for the
                    potential track and build the window around the predicted position. */

                T = RETURNS(CONTACT,3) - POT_TRACKS(IDX,3)
                POS(1) = POT_TRACKS(IDX,22) + T * POT_TRACKS(IDX,23) +
                    POT_TRACKS(IDX,24) * T * T /2
                POS(2) = POT_TRACKS(IDX,25) + T * POT_TRACKS(IDX,26) +
                    POT_TRACKS(IDX,27) * T * T /2

                /* COPY THE COVARIANCES TO THE P ARRAY. (LENGTH EQUATION) */

                DO I = 4 TO 21
                  P(I) = POT_TRACKS(I)
                END DO

                R(1) = P(4)+T*(P(7)+P(5))+T*T*(P(10)/2+P(8)+P(6)/2)+T*T*T/2*(P(11)+P(9))+
                  T*T*T*T/4*P(12)

                R(2)=P(13)+T*(P(16)+P(14))+T*T*(P(19)/2+P(17)+P(15)/2)+T*T*T/2*(P(20)+P(18))+
                  T*T*T*T/4*P(21)

                R_VEL_CAL = ABS ( (POS_RET(1) - POS(1) ) / T )
                B_VEL_CAL = ABS ( (POS_RET(2) - POS(2) ) / T )

                /* Check the speed thresholds */
                IF ( (R_VEL_CAL >= V_MIN(1) ) AND
                  (R_VEL_CAL <= V_MAX(1) ) AND
                  (B_VEL_CAL >= V_MIN(2) ) AND
                  (B_VEL_CAL <= V_MAX(2) ) ) THEN

                  UPDATE_FLAG = 1
                END IF
              ENDIF
            END IF
            IF (UPDATE_FLAG > 0) THEN
              /* Build the window and calculate the statistical distance */
              DO I=1 TO 2
                MAX_VALUE = POS_RET(I) - POS(I) - T * V_MAX(I)
                MIN_VALUE = POS_RET(I) - POS(I) - T * V_MIN(I)
                dij(I) = AMAX1(MAX_VALUE,0)+AMAX1(-MIN_VALUE, 0)
              END
              dij(1) = dij(1)/( SQRT (PARAM(11)) + R(1) )
              dij(2) = dij(2)/( SQRT (PARAM(12)) + R(2) )
              Dij = SQRT ( dij(1)*dij(1) + dij(2)*dij(2) )

              /*  Check if the calculated statistical distance is less then the preset
                  threshold */

              IF (Dij <= LAMBDA) THEN
                IF (MIN_Dij = -1.0 OR Dij < MIN_Dij) THEN

                  /*  The track with the minimum Ds value is associated with the return */
                  MIN_Dij = Dij
                  ASSOC_ARRAY(CONTACT) = IDX
                END IF
              END IF
            END IF
          END DO
        END IF
      END DO
```

P148371.PDF [Page: 49 of 61]

3.3.5 UPDATE_TRACK Routine Design

3.3.5.1 Description

This routine updates the potential tracks with the associated returns. The estimated covariances and the positions are calculated and written in the POT_TRACKS array. The new measurements are also recorded in the POT_TRACKS array.

This routine deletes the potential tracks which do not receive hits for this scan.

This routine is called by INIT1 module only.

3.3.5.2 Inputs

REAL RETURNS(0:LIM_RET, 15):
This array contains the information for all the returns from the current scan. The RETURNS array is explained in Sec. 3.1.1 as A array.

REAL NEW_TRACKS(LIM_RET,7):
Where LIM_RET is user-defined parameter. This is a temporary array used to pass the information about the new tracks to the ADD_TRACKS routine.

INTEGER NEW_NUM:
This variable keeps the number of new tracks created in this scan

3.3.5.3 Outputs

REAL POT_TRACKS(N*LIM_RET,40):
This table keeps the information about the potential tracks created by the initiation process. POT_TRACKS array is described in section 4.3 in detail.

3.3.5.4 Global Data

REAL ASSOC_ARRAY(LIM_RET):
This array is used to store the results of the association process. Association process associates either

- current returns to old returns or;
- current returns to potential tracks.

P148371.PDF [Page: 50 of 61]

The indexing of the ASSOC_ARRAY is the same as RETURNS array indexing. (i.e., ASSOC_ARRAY(1) keeps information about the first element in RETURNS array).

In the case of the return/old_return association process, this array keeps the index values of the associated old returns in the OLD_RET table.

If any location of the ASSOC_ARRAY contains a -1, this means that the corresponding return is not associated with any of the previous returns stored in OLD_RET table.

REAL POT_TRACKS(N*LIM_RET,40):
This table keeps the information about the potential tracks created by the initiation process. POT_TRACKS array is described in section 4.3 in detail.

REAL PARAM(2), PARAM(11), PARAM(12):
This routine accesses the 2nd ,the 11th and the 12th locations of the PARAM array to retrieve the scan rate, the range and the bearing measurement variances respectively.

LIM_RET:
The maximum number of returns per scan that is processed by ATP routine.

3.3.5.5 Local Data

REAL COVEST(2,3,3):
Temporary array used to store the covariance estimates.

REAL POSEST(2,3):
Temporary array used to store the estimated position.

REAL POS1(2):
keeps the range and bearing for the first set of measurements.

REAL POS2(2):
keeps the range and bearing for the second set of measurements.

REAL POS3(2):
keeps the range and bearing for the third set of measurements.

REAL V_MIN(2), V_MAX(2):
These arrays keep the minimum and the maximum speed thresholds for range and bearing

P148371.PDF [Page: 51 of 61]

REAL SIGMA(2):
contains the measurement variances for the range and the bearing.

REAL COEF(2):
contains the squared value of the measurement variances.

REAL T3:
keeps the time of the third (most recent) measurement

REAL T:
keeps the time difference between the second and the third set of measurements divided by 2

REAL A1,A2,A3:
Temporary variables used in the calculation of the position estimation

INTEGER I,J,K,CONTACT,TRK_ID,INDEX, TRK_IDX:
These variables are used for loop and array indexing.

3.3.5.6 Main Body of the Routine

```
/* Initialize the measurement variances for range and bearing */

  SIGMA(1) = SQRT (PARAM(11))
  SIGMA(2) = SQRT (PARAM(12))

  /*  First set the update flag for the potential track to 0.0. This field is set to 1.0
      only when there is an update in this scan */

  DO I=1 TO POT_TRACKS(0,1)
    POT_TRACKS(I,38) = 0.0
  END DO

  /* set the minimum and the maximum speeds */

  V_MIN(1) = PARAM(7)
  V_MAX(1) = PARAM(6)
  V_MIN(2) = PARAM(7)/150000 ! 150000 is the maximum detection range
  V_MAX(2) = PARAM(6)/6000 ! 6000 is the minimum detection range

  DO I=1 TO 2
    COEF(I) = SIGMA(I)*SIGMA(I)/2
  END DO

  DO CONTACT=1 TO RETURNS(0,1)

    IF (ASSOC_ARRAY(CONTACT) <> -1) THEN

      TRK_ID = ASSOC_ARRAY(CONTACT)

      IF (POT_TRK(TRK_ID,38) = 1.0) THEN

        /*  The update flag shows that the track has been updated this scan, therefore if
            there are other returns associated with this track a split must be created */
```

P148371.PDF [Page: 52 of 61]


```
T = (RETURNS(CONTACT,3) - POT_TRK(TRK_ID,30))/2
T_1 = POT_TRK(TRK_ID,30)
T_2 = RETURNS(CONTACT,3)

M_1(1) = POT_TRK(TRK_ID,28)
M_1(2) = POT_TRK(TRK_ID,29)

M_2(1) = RETURNS(CONTACT,4)
M_2(2) = RETURNS(CONTACT,5)

R_VEL_CAL = ABS ( (M_2(1)-M_1(1)) / (2*T) )
B_VEL_CAL = ABS ( (M_2(2)-M_1(2)) / (2*T) )

/* Check the speed thresholds */
IF ( (R_VEL_CAL >= V_MIN(1) ) AND
  (R_VEL_CAL <= V_MAX(1) ) AND
  (B_VEL_CAL >= V_MIN(2) ) AND
  (B_VEL_CAL <= V_MAX(2) ) ) THEN

  /* Write the track into the table */
  TRK_IDX = POT_TRK(0,1)+1
  POT_TRK(TRK_IDX,1) = 1 !VALID FLAG
  POT_TRK(TRK_IDX,3) = RETURNS(CONTACT,3) !TIME

  /* POT_TRACKS array location 4-12 contain covariance estimate for range */
  /* POT_TRACKS array location 13-21 contain covariance estimate for bearing */

  /* Clear the table before writing the track */
  DO INDEX = 4 TO 27
    POT_TRK(TRK_IDX,INDEX) = 0.0
  END DO

  POT_TRK(TRK_IDX,4) = COEF(1)
  POT_TRK(TRK_IDX,13) = COEF(2)
  POT_TRK(TRK_IDX,8) = 2*COEF(1)/(4*T*T)
  POT_TRK(TRK_IDX,17) = 2*COEF(2)/(4*T*T)

  /* POT_TRACKS array location 22-24 contain position vector for range */
  /* POT_TRACKS array location 25-27 contain position vector for bearing */
  INDEX = 22
  DO I=1 TO 2
    A0 = (M_1(I) + M_2(I))/2
    A1 = (M_2(I) - M_1(I))/(2*T)
    POT_TRK(TRK_IDX,INDEX) = A0+A1*T
    INDEX = INDEX +1
    POT_TRK(TRK_IDX,INDEX) = A1
    INDEX = INDEX+2
  END DO
  /*   POT_TRACKS array location 28-32 contain Range, Bearing, Time, X-pos and Y-pos
       for the measurement 1 */

  POT_TRACKS(TRK_IDX,28) = M_1(1)
  POT_TRACKS(TRK_IDX,29) = M_1(2)
  POT_TRACKS(TRK_IDX,30) = T_1
  POT_TRACKS(TRK_IDX,31) = POT_TRK(TRK_ID,31)
  POT_TRACKS(TRK_IDX,32) = POT_TRK(TRK_ID,32)

  /*   POT_TRACKS array location 33-37 contain Range, Bearing, Time, X-pos and Y-pos
       for the measurement 2 */

  POT_TRACKS(TRK_IDX,33) = M_2(1)
  POT_TRACKS(TRK_IDX,34) = M_2(2)
  POT_TRACKS(TRK_IDX,35) = T_2
  POT_TRACKS(TRK_IDX,36) = RETURNS(CONTACT,1)
  POT_TRACKS(TRK_IDX,37) = RETURNS(CONTACT,2)

  /*   POT_TRACKS array location 38 contains update flag (0 if the potential track
       is not updated this scan*/
```

P148371.PDF [Page: 53 of 61]


```
      POT_TRACKS(TRK_IDX,38) = 1.0

      /*   location 39 is empty */

      /* Increment the number of the potential tracks */

      POT_TRK(0,1) = POT_TRK(0,1)+ 1

      /* Remove the contact from further consideration */

      RETURNS(CONTACT,10) = -999.99

    END IF

  ELSE

    T = (RETURNS(CONTACT,3) -  POT_TRACKS(TRK_ID,35))/2
    POS3(1) = RETURNS(CONTACT,4) /* Range */
    POS3(2) = RETURNS(CONTACT,5) /* Bearing */
    T3 = RETURNS(CONTACT,3)-POT_TRK(TRK_ID,35)
    POS2(1) = POT_TRACKS(TRK_ID,33)
    POS2(2) = POT_TRACKS(TRK_ID,34)
    POS1(1) = POT_TRACKS(TRK_ID,28)
    POS1(2) = POT_TRACKS(TRK_ID,29)

    DO I=1 TO 2
      COVEST(I,1,1) = SIGMA(I)
      COVEST(I,1,2) = 0.0
      COVEST(I,1,3) = -2/(T*T)
      COVEST(I,2,1) = 0.0
      COVEST(I,2,2) = 1/(2*T*T)
      COVEST(I,2,3) = 0.0
      COVEST(I,3,1) = -2/(T*T)
      COVEST(I,3,2) = 0.0
      COVEST(I,3,3) = 6/(T*T*T*T)

      A0 = POS2(I)
      A1 = (POS3(I)-POS1(I))/(2*T)
      A2 = (POS1(I)-2*POS2(I)+POS3(I))/(T*T)

      POSEST(I,1) = A0+A1*T3+A2*T3*T3/2
      POSEST(I,2) = A1+A2*T3
      POSEST(I,3) = A2
    END DO
    POT_TRK(TRK_ID,3) = RETURNS(CONTACT,3) !TIME

    /* Range Covariance Estimates, Locations 4-12 , Bearing Covariance Estimates,
    Locations 13-21 */
    INDEX = 4
    DO I=1 TO 3
      DO J=1 TO 3
        DO K=1 TO 3
          POT_TRACKS(TRK_ID,INDEX) = COVEST(I,J,K)
          INDEX = INDEX+1
        END DO
      END DO
    END DO

    /*  Calculate the position, velocity and acceleration estimates */

    DO I=1 TO 2
      DO J=1 TO 3
        POT_TRACKS(TRK_ID,IDX) = POSEST(I,J)
        INDEX = INDEX+1
      END DO
    END DO

    /* Set the flag showing that the track is updated this scan */
```

P148371.PDF [Page: 54 of 61]


```
          POT_TRACKS(TRK_ID,38)  = 1.0

          /*  Shift the recorded measurements from the previous scan by 1 in POT_TRACKS Table
              */

          DO I = 28 TO 32
            J = IDX+5
            POT_TRACKS(TRK_ID,I) = POT_TRACKS(TRK_ID,J)
          END DO

          /* Now store the new measurements */
          POT_TRACKS(TRK_ID,33) = RETURNS(CONTACT,4)
          POT_TRACKS(TRK_ID,34) = RETURNS(CONTACT,5)
          POT_TRACKS(TRK_ID,35) = RETURNS(CONTACT,3)
          POT_TRACKS(TRK_ID,36) = RETURNS(CONTACT,1)
          POT_TRACKS(TRK_ID,37) = RETURNS(CONTACT,2)

          /* Remove the return from further consideration */
          RETURNS(CONTACT,10) = -999.99
        END IF
      END IF
    END DO

    /*  Do the house_keeping functions: If the potential track is not updated this scan,
        delete it from the table  */

    DO I=1 TO POT_TRACKS(0,1)

      /* Mark the tracks for deletion */
      IF (POT_TRACKS(I,38) .NE. 1.0) THEN

        INDEX=INDEX+1
        POT_TRACKS(I,1) = 0.0

      END IF
    END DO

    /* Now actually delete the tracks*/

    TRK_ID = 1
    DO WHILE (INDEX > 0)
      IF (POT_TRK(TRK_ID,1) = 0.0) THEN

        /* Shift the tracks up */
        DO I=TRK_ID+1 TO POT_TRK(0,1)
          K=I-1
          DO J=1 TO 40
            POT_TRK(K,J) = POT_TRK(I,J)
          END DO
        END DO
        INDEX = INDEX -1
        POT_TRK(0,1) = POT_TRK(0,1) -1
      ELSE
        TRK_ID = TRK_ID + 1
      END IF
    END DO
```

P148371.PDF [Page: 55 of 61]

4 ATP GLOBAL DATA

4.1 Baseline - ATP Modifications

REAL OLD_RET(0:50,50,10):
This array keeps the returns from the previous scan. It has the same structure as the B array of the original ATP program.

Note that OLD_RET array has been defined in the FRAME module as array B in the original version of the ATP program. The above declaration will be saved in an include file to be included when the original version of the ATP is built.

4.2 Hough Transform Track Initiation

Following are the new global data of the ATP program when for the Hough Transform Track Initiation. These data will be stored in an include file.

INTEGER COUNT:
This global variable is set to 0 after every initiation process. This variable is used to store the number of scans during the waiting period.

REAL OLD_RET(0:MAX_NR_OF_SCANS*LIM_RET, 11):
This array keeps the return from a number of previous scans. It has the same structure as A array explained in Sec. 3.1.1.

REAL LOOK_UP(4,0:8):
This array is used in the grouping of the parallel tracks in the histogram using the nth order averaging methodology. LOOK_UP array is initialized with the following values.

LOOK_UP(1,0)=1
LOOK_UP(1,1)=1

LOOK_UP(2,0)=2
LOOK_UP(2,1)=11
LOOK_UP(2,2)=12

LOOK_UP(3,0)=4
LOOK_UP(3,1)=111
LOOK_UP(3,2)=112
LOOK_UP(3,3)=121
LOOK_UP(3,4)=122

P148371.PDF [Page: 56 of 61]

LOOK_UP(4,0)=8
LOOK_UP(4,1)=1111
LOOK_UP(4,2)=1112
LOOK_UP(4,3)=1121
LOOK_UP(4,4)=1122
LOOK_UP(4,5)=1211
LOOK_UP(4,6)=1212
LOOK_UP(4,7)=1221
LOOK_UP(4,8)=1222

Note that the row number corresponds to the order of the averaging.

REAL VECT(LIM_RET*MAX_NR_OF_SCANS):
This array keeps the grouping information for each return. The value of each location is one of the defined values in the LOOK_UP array. (i.e, If we are processing the 2nd order averaging and the return number 3 has identified as in group 11, then the 3rd location of VECT array contains the value 11.

REAL PI:
Literal value for 180 degrees

INTEGER LIM_RET:
The number of maximum returns per scan that ATP can process.(Initialized to 50)

REAL HIST(0:LIM_RET*MAX_NR_OF_SCANS):
The histogram array(i.e., bin), rows corresponding to the returns. This array keeps the rho values for each return with different theta value.

INTEGER MAX_NR_OF_SCANS:
The literal value for the maximum number of scans of waiting period before applying the Hough Transform Technique.

4.3 Logic_based Track Initiation

Following are the new global data of the ATP program when for the Logic_based Track Initiation. These data will be stored in an include file.

REAL POT_TRACKS(0:N*LIM_RET,40):
N will be defined in a later stage. This array keeps information about the potential tracks until they are confirmed or deleted.

(0,1): The number of potential tracks

P148371.PDF [Page: 57 of 61]

(i,1):	Valid Flag: 1 if the track is valid, 0 otherwise
(i,2):	empty (can be used as track id)
(i,3):	Time of last update
(i,4)-(i,12):	Range Covariance Estimate
(i,13)-(i,21):	Bearing Covariance Estimates
(i,22)-(i,24):	Range position, velocity and acceleration estimates
(i,25)-(i,27):	Bearing position, velocity and acceleration estimates
(i,28):	Range information for measurement 1
(i,29):	Bearing information for measurement 1
(i,30):	Time of measurement 1
(i,31):	x-coordinate of measurement 1
(i,32):	y-coordinate of measurement 1
(i,33):	Range information for measurement 2
(i,34):	Bearing information for measurement 2
(i,35):	Time of measurement 2
(i,36):	x-coordinate of measurement 2
(i,37):	y-coordinate of measurement 2
(i,38):	update flag; 1 if the track receives a hit this scan, 0 otherwise

REAL ASSOC_ARRAY(LIM_RET):
This array is used to store the results of the association process. Association process associates either

- current returns to old returns or;
- current returns to potential tracks.

The indexing of the ASSOC_ARRAY is the same as RETURNS array indexing. (i.e., ASSOC_ARRAY(1) keeps information about the first element in RETURNS array).

In the case of the return/old_return association process, this array keeps the index values of the associated old returns in the OLD_RET table.

In the case of the association process of the potential tracks/returns, it keeps the index values of the associated potential tracks in the POT_TRACKS array.
REAL ASSOC_ARRAY(LIM_RET):

If any location of the ASSOC_ARRAY contains a -1, this means that the corresponding return is not associated with any of the previous returns stored in OLD_RET table.

P148371.PDF [Page: 58 of 61]

REAL OLD_RET(0:LIM_RET,11):
This array keeps the returns from the previous scans. The location (0,1) keeps the number of the old_returns. It has the same structure as A array explained in Sec 3.1.1.

INTEGER CONTACT_ASSOC:
This variable is used as a flag for the association process. It indicates the association between returns from the current scan and the returns from the previous scan. Initialized to 1.

INTEGER TRACK_ASSOC:
This variable is used as a flag for the association process. It indicates the association between returns from the current scan and the returns from the previous scan. Initialized to 2.

INTEGER LIM_RET:
The maximum number of returns per scan that is processed by ATP routine.(Initialized to 50)

INTEGER COUNT:
This global variable is set to 0 after every initiation process. This variable is used to store the number of the processed scans during initialization.

INTEGER MAX_NR_OF_SCANS:
This variable keeps the maximum number of scans to be processed by initiation routine before the track confirmation.

5 ATP DATA FILES

5.1 CASE.DAT File

The CASE.DAT file remains unchanged.

5.2 Radar Data File

The performance of the track initiation routines implemented in this phase of the track project will be investigated using the radar plots collected at Cold Lake, Alberta during a trial involving scenarios of six CF-18 aircraft flying in close formation.

UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM
(highest classification of Title, Abstract, Keywords)

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (the name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Establishment sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Thomson-CSF Systems Canada Inc.
18 Auriga Drive
Nepean, Ontario K2J 7T9

2. SECURITY CLASSIFICATION (overall security classification of the document. including special warning terms if applicable)

UNCLASSIFIED

3. TITLE (the complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S,C or U) in parentheses after the title.)

Multi-Target Tracking - Phase II
Task 2 - Detail Design Report. (U)

4. AUTHORS (Last name, first name, middle initial)

Kalaycioglu, Banu
Kurkjian, Azaduer

5. DATE OF PUBLICATION (month and year of publication of document)

December 1992

6a. NO. OF PAGES (total containing information. Include Annexes, Appendices, etc.)

54

6b. NO. OF REFS (total cited in document)

6

7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Technical Report

8. SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development. Include the address.)

Defence Research Establishment Ottawa
3701, Carling Ave
Ottawa, Ontario, K1A 0K2

9a. PROJECT OR GRANT NO. (if appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant)

Project 041LO

9b. CONTRACT NO. (if appropriate, the applicable number under which the document was written)

W7714-2-9620/01-ST

10a. ORIGINATOR'S DOCUMENT NUMBER (the official document number by which the document is identified by the originating activity. This number must be unique to this document.)

G0649C-02

10b. OTHER DOCUMENT NOS. (Any other numbers which may be assigned this document either by the originator or by the sponsor)

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

(X) Unlimited distribution
( ) Distribution limited to defence departments and defence contractors; further distribution only as approved
( ) Distribution limited to defence departments and Canadian defence contractors; further distribution only as approved
( ) Distribution limited to government departments and agencies; further distribution only as approved
( ) Distribution limited to defence departments; further distribution only as approved
( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availabilty (11). However, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected.)

Same as in 11

UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM

UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is h desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), or (U). It is not necessary to include here abstracts in both offical languages unless the text is bilingual).

The Radar Division of DREO has developed a Multi-Target Trackin software testbed referred to as the Automatic Tracking Program (ATP package, in order to process radar data in real time. The objective o the first phase of the project was to study and assess the ATP packag in order to provide in-depth understanding of the limitations an problem area of the package. The objective of this second phase is t investigate alternative track initiation techniques based on Houg transform and sequential processing. These new techniques can b simulated and tested in the context of the ATP software environment The purpose of the Task 2 is to give the detailed design specification of the new track initiation modules described in the Task 1 report.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could b helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be sele from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it is not possible select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Multi-Target Tracking, Automatic Tracking Testbed, Track Initiatio Techniques, Hough Transform, Logic Based, Sequential Processing.